UCRL-TR-211020

LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

Characterizing Network Services through Cluster-Set Variations

A. Bartoletti, N. Tang

April 1, 2005

Disclaimer

This document was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor the University of California nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or the University of California, and shall not be used for advertising or product endorsement purposes.

This work was performed under the auspices of the U.S. Department of Energy by University of California, Lawrence Livermore National Laboratory under Contract W-7405-Eng-48.

Lawrence Livermore National Laboratory
[UCRL-TR-211020]

Characterizing Network Services through Cluster-Set Variations

Tony Bartoletti
Nu Ai Tang

2004

Work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract W-7405-Eng-48.

Characterizing Network Services through Cluster-Set Variations

Tony Bartoletti
Nu Ai Tang

ABSTRACT

Common Internet services can be reliably distinguished based solely upon the locations of clusters in traffic-based features (ratios of inbound to outbound packets, ratios of packets to payloads, etc.) This capability has value in revealing the nature of "hidden" (tunneled) services and in detecting anomalous changes to known services. We provide measures of session capture volumes sufficient to make confidence-level assertions regarding "unknown" services, and outline a throughput system for providing alarms for service anomalies.

BACKGROUND

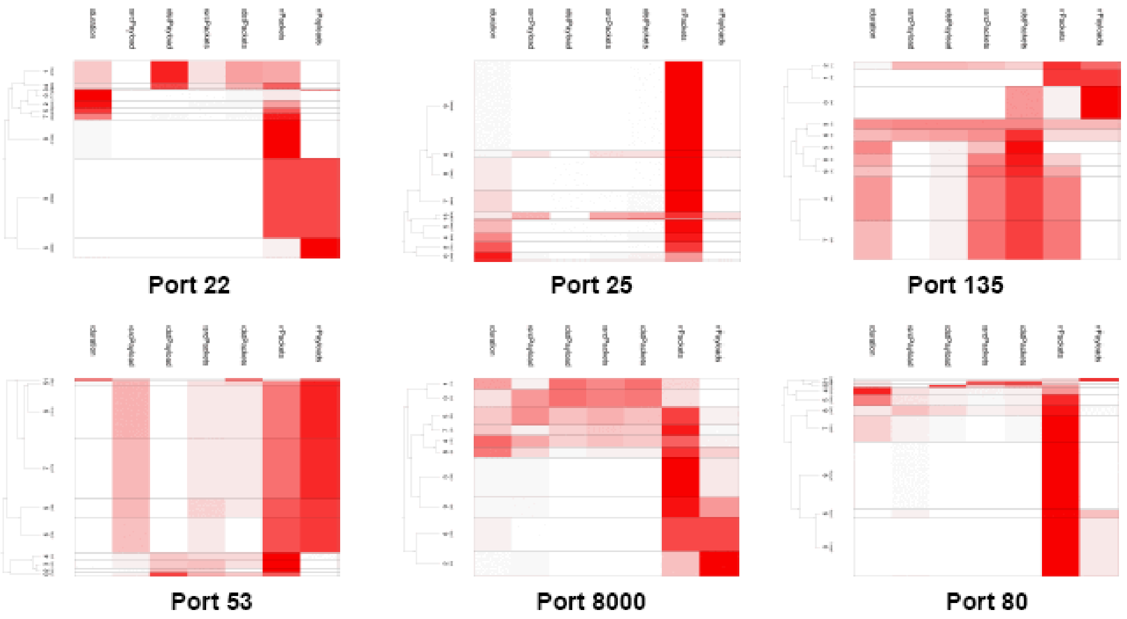
Most network intrusion detection systems (IDSs) operate by monitoring network traffic in the most intrusive manner. Typically, the actual packet contents (payloads) are examined, in order to identify "known-harmful" strings, embedded commands intended to overflow buffers of vulnerable services and execute unauthorized codes, yielding control of the service to the attacker. Although there are variants of such IDS that involve "fuzzier" heuristics, most will fail to identify truly new forms of exploitation (for which the characteristic "exploit" code is not yet a known-string or derivative pattern). Moreover, once the exploit has taken hold, subsequent unauthorized traffic may well "look like" legitimate traffic (at least, on a session-by-session basis) and thus provide no further indication that the system has been compromised. Finally, many services are often "tunneled" within another encrypted protocol (e.g., ssh), rendering the actual packet contents opaque to traditional IDS.

Therefore, service characterizations that can be made entirely upon the elements of traffic that must remain un-encrypted (the packet-routing, packet count and packet size information) provide a valuable adjunct to traditional IDS.

Prior examination of traffic features performed over session summaries of LLNL CPP network capture revealed distinctive clustering patterns for different services (telnet, ftp, http, smtp, to name a few) as well as distinctive patterns on the encrypted SSH traffic, when partitioned according to the major destination machines servicing that protocol. In this latter case, it is surmised that individual SSH servers were largely tunneling a fixed and distinct service or activity.

This project set out to build upon these observations by providing a methodology for distinguishing services in an operational environment. This requires both determining the number of records one must capture in order that the cluster set formed would tend to reach a stable set of proportions, and also a "distance-like metric" between clustering patterns in order to make judgments on the relative closeness of patterns. Cluster sets of varying size were also needed in order to demonstrate that one could test hypotheses regarding services with variable levels of confidence.

The following graphic depicts the clustering characteristics of six selected ports, produced prior to this study by Imola Fodor on behalf of CIAC data analysis.



Port 22

Port 25

Port 135

Port 53

Port 8000

Port 80

Port Behavior Clusters (via CLUTO)

It is clear by mere observation that each service supports a range of different session qualities, and that different services display very different clusters of such behaviors.

In each of the six cluster graphs above, the 7 columns represent the 7 features we had chosen to record or calculate for each "session". These features (from left to right) are

- Session Duration (in seconds)
- Source Payload Bytes (sent to port)
- Destination Payload Bytes (received from port)
- Source Packet Count (sent to port)
- Destination Packet Count (received from port)
- Ratio of Packets (sent/received)
- Ratio of Payloads (sent/received)

For each port, many thousands of such records were collected, and then clustered using the CLUTO[1] clustering tool, asking for 10 clusters on each occasion. The 10 resulting clusters are depicted as rows in the cluster graphs. The thickness (height) of a given row is proportional to the number of records that landed in that particular cluster, and for each row, the color intensity for each column depicts the mean value of the corresponding feature, over the records in that cluster, with red indicating a high value, and white a low value. For instance, the top row of the port 53 cluster set is the only cluster with a very high average session duration (leftmost column), but has very few records. This may reflect DNS update activity, in contrast to the more varied DNS queries being serviced.

Put another way, each session record is treated as a 7-dimensional vector, and the colors in a given row reflect the values of the centroid-vector for the vectors in that cluster. For the purposes of this study, we have no particular interest in individual session records, except as they contribute to the formation of the clusters and the determination of the resulting cluster centroids. Therefore, I will generally use the term "**cluster set**" to refer to the set of 10 centroid vectors determined by the clustering process.

THE DATA

The study we conducted used data captured at LLNL by a "Niksun" session-summary product, as part of the DOE Cooperative Protection Program (CPP). The data covered the month of February 2004, and amounted to about 500 Gb uncompressed data, representing approximately one billion "sessions" (80% are actually TCP probes and not true sessions, and probably 90% of the remainder are HTTP "get" commands induced when web pages are processed). The Niksun sensor makes no attempt to distinguish internal from external services, and thus the sessions captured may have originated from either external or internal clients. Moreover, "session summaries" are really session traffic summaries. The actual packet payloads are not examined. Each session summary consists of a set of identifiers and statistics, which included:

a. Session Duration in seconds
b. Protocol (TCP, UDP, etc)
c. Source IP Address
d. Source IP port
e. Dest IP Address
f. Dest IP port
g. SrcPackets (Total packets sent by Source)
h. DstPackets (Total packets returned by Dest)
i. SrcPayload (Total payload from Source, excludes packet headers)
j. DstPayload (Total payload from Dest, excludes packet headers)

To ensure that we were dealing with "true" sessions and not with hostile (unanswered) probes, we reduced the data by requiring the conjunction

```
SrcPackets > 0
DstPackets > 0
SrcPayload > 0
DstPayload > 0
```

This restriction leaves approximately 200,000,000 "real" sessions for the month. We also had to eliminate about 0.3 % of these records, as they were reported by the capture tool as having a negative duration. We dispensed with the source and destination IP addresses and the source port, as these were unnecessary for our purposes.

Finally, we partitioned all of the records according to destination port and protocol (hence, by “service”) and reduced the records to 7-dimensional vectors of the form:

<Duration,SrcBytes,DstBytes,SrcPackets,DstPackets,ByteRatio,PacketRatio>

with the caveat that ALL values were converted to log(1+value) to provide better separation in the face of extreme values.

THE CONCEPT

Typical network sessions with specific services (Telnet, Web/HTTP, FTP, Sendmail, etc) exhibit different traffic characteristics. For instance, a typical Telnet session may have long duration (the user remains logged-in for a long time), and most often, the client (source IP) sends relatively few packets or bytes, consisting mostly of shell-based commands such as "ls", while the server (destination IP) may return hundreds of packets or bytes in return, such as a very long directory listing resulting from the aforementioned "ls" command. In contrast, HTTP (Web) "sessions" are typically very short (each "fetch" of a single page, or page component is a complete "session" in the TCP sense.)

However, any individual session, in almost any service, can be far from the "typical" session for that service. Thus, rather than attempt to characterize individual sessions (or characterize a service by calculating its “average” session), we claim a more stable position by arguing that the *distribution* of session behaviors seen for each type of service are strongly characteristic of that service. Hence, upon reduction of each evident session to a record of standard features, we apply clustering methods to a large population of such sessions, and consider the set of resulting centroids (the “averaged” session for each cluster in the cluster set) to represent our signature of the service in question.

When given a large body of records of standard features, such as

<duration,bytes-in,bytes-out,packets-in,packets-out>

clustering methods employ various heuristics to group these records into individual "clusters", such that records having very similar feature values end up in the same cluster, while those that differ greatly in one or more features end up in different clusters. One can arbitrarily demand that a clustering method produce (say) ten clusters, in which case an attempt is made to divide-up the records in a way that will produce the "best" (most distinct) ten clusters. This is the approach we have taken (specifically, the method of repeated bisections in the formation of 10 clusters) so that we would be able to compare the clustering results across different populations of services and session behaviors.

Given a sufficiently large sample from a population of session records for a given service, we expect (under the demand of, say, 10 clusters) that the resulting cluster set tends to stabilize. That is, a subsequent sample from the same population would result in a similar cluster set. Once a reasonable numeric measure of the disparity between two cluster sets had been established, we could examine the degree of this stabilization as a function of sample size, and confirm our hypothesis that this derived "distance" between cluster-sets will remain significantly larger when the comparison is between populations from disparate services (e.g, Telnet versus FTP) than it is between samples from the same population.

Our goal is then to quantify this relation between sample size and expected distance, in support of "likelihood assertions" regarding newly observed service behaviors.

THE METHODOLOGY

To begin the study, we partitioned the session records by day, according to destination port. This gave us a sense of the number of sessions one could expect, for each service, in a given time interval. Foremost, it served to help us further restrict the range of services we would study, as many of these were not evident in sufficient number to warrant a strong statistical treatment. This left us to study the following 6 services:

Port 21	FTP
Port 22	SSH
Port 25	SMTP (Sendmail)
Port 80	HTTP (web services)
Port 110	POP (Post Office Protocol)
Port 443	HTTPS (secure web)

For these remaining services, records were aggregated in sets of size 1000, 2000, 4000, and 8000 (and occasionally in other sizes.) The volume of HTTP (web) traffic was such that only about 1 day's worth of sessions was sufficient to produce 50 or so sets in each size, while other services required a week or more of the session summaries.

Note: Originally, we desired many sets of each size in {1000,2000,4000,8000} in order to investigate the degree to which stability would increase with increasing sample sizes. Although we conducted sufficient tests to confirm this behavior, we focused most of our attention on the largest size record collections, size 8000, in order to establish the effectiveness of the discrimination method. For most of the following discourse, consider each data set to contain 8000 records.

For each set, clustering was applied via CLUTO to generate 10 clusters, and the consequent set of 10 centroids was recorded, along with the number of records in each corresponding cluster. These numbers were applied as "weights" when conducting the cluster-set comparisons described below.
The 10 cluster-set centroids serve to characterize the entire set of records that were subject to the clustering (and hence, serve to characterize the service, if all records were from a single service.)

One is then interested in knowing how much variance to expect when performing clustering on two separate populations of records, each population (ostensibly) representing sessions of the same service type (e.g., both are records of Telnet sessions.) This requires a measure of the "distance" between two cluster-sets. The distance function we employed is described here.

> Let **P** and **Q** be two populations of records possessing **f** features from the same schema (common positional fields measure the same feature). Let p_i be the i-th record in set **P**.
>
> Let **CP** = {$\mathbf{CP}_k$}, for k = 1…10 be the partitioning of **P** into 10 clusters according to the CLUTO default settings (method of repeated bi-sections).
>
> Let **cP =** {$\mathbf{cP}_k$} be the set of 10 centroids for the clusters of **CP**.
>
> Likewise, let **cQ =** {$\mathbf{cQ}_k$} be the set of 10 centroids for the clusters of **CQ**.

The "cluster-set distance" we seek is D(**P**,**Q**) = | **cP** - **cQ** |. The question remained, how to define | **cP** - **cQ** |.

Naively, one would like to define this cluster-set distance simply by summing the pairwise "vector-differences" | $\mathbf{cP}_k - \mathbf{cQ}_k$ |. Unfortunately, there is no a-priori way to establish that the k-th cluster $\mathbf{CP}_k$ of **P** *corresponds* to the k-th cluster $\mathbf{CQ}_k$ of **Q**. Indeed, under many clustering schemes, a tool may produce a different clustering order when given the same records in an alternate sequence. Effectively, one is left with two sets **cP** and **cQ** of 10 centroids each, each set unordered.

It is therefore most natural to define the distance between **CP** and **CQ** to be the minimal distance possible that can be generated by pairing each $\mathbf{cP}_j$ to $\mathbf{cQ}_k$, in sets of 10 pairs where |{j}| = |{k}| = 10. For cluster-sets having 10 clusters, this represents a daunting 10! = 3,628,800 possible sets, each of 10 pairs, to examine exhaustively.

Fortunately, there is a tractable solution to this minimization problem via the Hungarian matching algorithm, with order $O(n^{1.5})$ complexity[2]. We create a 10x10 matrix of the 100 possible individual vector differences

$$M = [\ M_{i,j}\ ] = [\ |\ \mathbf{cP}_i - \mathbf{cQ}_j\ |\ ]$$

and the Hungarian algorithm returns the set of 10 pairs { (ix, jy) } indicating precisely the matching that will result in the minimal sum of vector differences. The corresponding sum of (Euclidean) vector differences was then calculated.

Having in hand a reasonable cluster-set distance function, D(P,Q), we needed to establish empirically the "typical" distance between cluster sets formed from a common service population, in comparison to the distance values derived when comparing cluster sets from different services. We conducted this examination by creating 10 to 20 sets of 8000 records from each service, generating their corresponding centroid sets, and then determining mean and standard deviation in the values D(P,Q).

The tables below show a result of these comparisons

Mean	80-6	21-6	22-6	25-6	443-6	110-6
80-6	1.3437	5.2379	3.9572	4.8454	2.6405	4.2445
21-6	*	0.827	6.4468	3.7639	5.2996	3.4534
22-6	*	*	1.0351	5.5877	4.0224	5.41
25-6	*	*	*	1.3731	4.982	4.9732
443-6	*	*	*	*	1.7452	4.668
110-6	*	*	*	*	*	1.1753

Stdv	80-6	21-6	22-6	25-6	443-6	110-6
80-6	0.7513	*	*	*	*	*
21-6	*	0.6317	*	*	*	*
22-6	*	*	0.4729	*	*	*
25-6	*	*	*	0.7516	*	*
443-6	*	*	*	*	0.7616	*
110-6	*	*	*	*	*	0.5098

Happily, we note that the "mean distance" between sets from the same service (values along the diagonal in the table of means) are everywhere significantly smaller than the off-diagonal values. One could employ these, together with the derived standard deviations, to argue (for instance) that SMTP (port 25) behavior appears to differ from HTTP (web) traffic behavior by

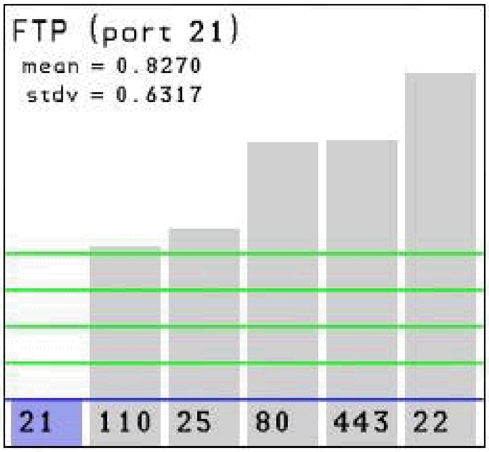
`(4.8454 - 1.3437)/0.7513 = 4.66` standard deviations.

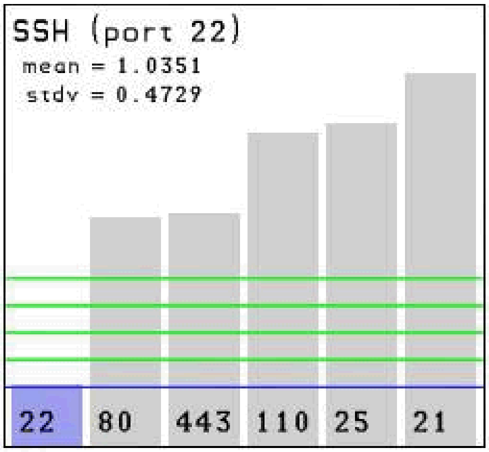
Hence, the likelihood of mistaking SMTP traffic for HTTP traffic, given 8000 records of each, is exceedingly small. In contrast, port 443 (HTTPS) differs from regular HTTP traffic by only

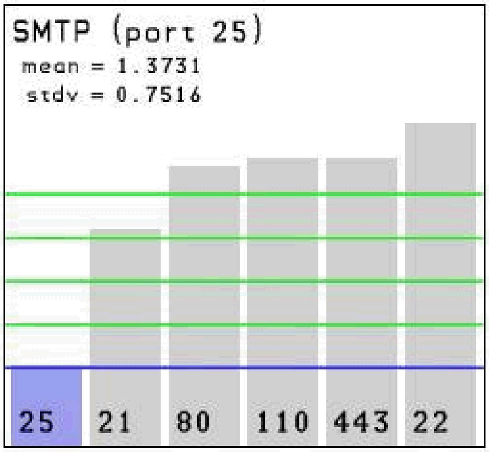
`(2.6405 - 1.3437)/0.7513 = 1.73` standard deviations.

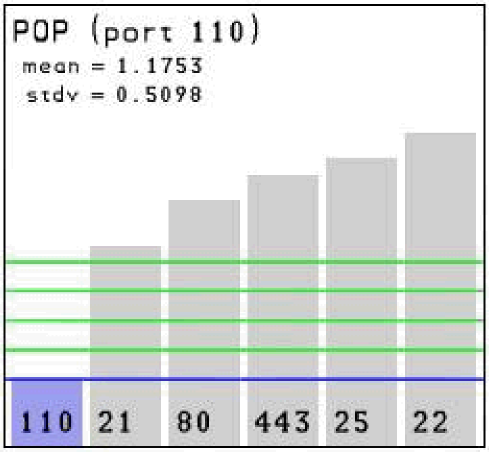
It is important to note that we are making these determinations entirely upon traffic statistics (ratios of packets to bytes, inbound versus outbound, etc) and without access to the packet content or a priori cognizance of the ports in question.

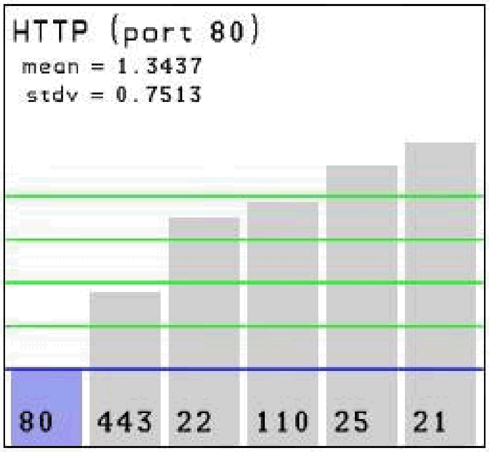
The chart below depicts each port, its mean and standard deviation in cluster set distance to that of clusters from the same and from different ports, and how far different ports lie in terms of standard deviations.

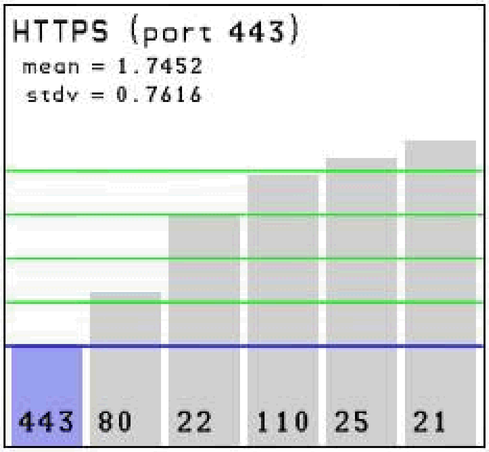

FTP (port 21)
mean = 0.8270
stdv = 0.6317
21
110
25
80
443
22


SSH (port 22)
mean = 1.0351
stdv = 0.4729
22
80
443
110
25
21


SMTP (port 25)
mean = 1.3731
stdv = 0.7516
25
21
80
110
443
22

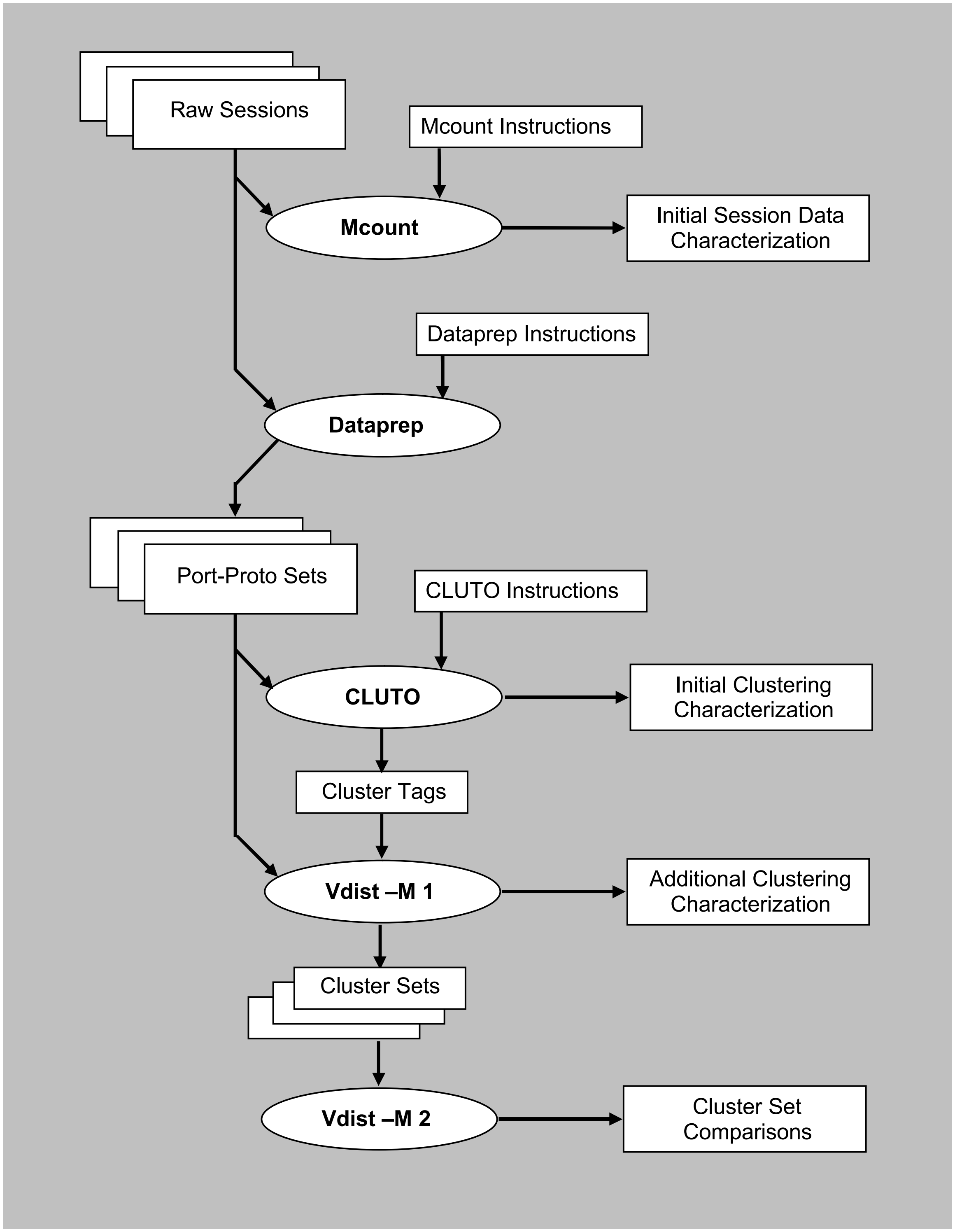

POP (port 110)
mean = 1.1753
stdv = 0.5098
110
21
80
443
25
22


HTTP (port 80)
mean = 1.3437
stdv = 0.7513
80
443
22
110
25
21


HTTPS (port 443)
mean = 1.7452
stdv = 0.7616
443
80
22
110
25
21

The chart below outlines the "throughput" system developed for both exploration and potential operational capabilities.

Process Flowchart



In the process flowchart, “MCount” accepts TCP session summary records, and provides (among other measures) the number of sessions by destination port. Having identified ports for which a sufficient number of sessions existed, session records were partitioned and passed to “DataPrep”, transforming the records into those containing the desired component values. These port-protocol sets each containing groupings of 8000 records.

Each port-protocol set was clustered using CLUTO, and then to a process “VDist –M 1” to determine the cluster-set centroids. Finally, “VDist –M 2” was called over multiple sets of centroids. In Mode 2, the Hungarian algorithm is employed to generate minimal cluster-set distances, further to conduct round-robin and KxK distance sets for calculation of means and standard deviations.

Future Work

There are many areas where refinement of measures should improve upon these results.

In terms of the raw data itself, the records clustered could have components formed through principal component analysis (PCA) rather than using the raw components.

In terms of the clustering operations, one could explore different clustering algorithms (agglomerative versus repeated-bisections), and vary the distance measure used in the clustering algorithm (the CLUTO default is cosine-distance.) One could also attempt to find an optimal number of clusters for this analysis (we selected 10 out of thin air.)

There would be additional value in determining the degree to which these cluster-set discrimination measures degrade as the number of available records is reduced.

Conclusion

We have demonstrated the ability to distinguish between common network services based entirely upon features extracted from traffic statistics. Such a capability may serve to identify “unlabeled” services operating on usual, or unusual service ports, and even to characterize a service tunneled with encryption through another dedicated service. Importantly, the fact that such characterization is possible in the absence of packet content inspection mitigates many privacy concerns that might otherwise hamper data sharing efforts.

References

[1] CLUTO, George Karypis, University of Minnesota, http://www.cs.umn.edu/~karypis/cluto

[2] http://ai.stanford.edu/~gerkey/tools/hungarian.html